JASPER REPORT - FILLING REPORTS

The main purpose of any reporting tool is to produce high quality documents. Report filling process helps reporting tool to achieve this by manipulating sets of data. The main input required for report-filling process is:

- **Report Template:** This is actual JasperReport file
- **Report Parameters** These are basically named values that are passed at the report filling time to the engine. We will discuss them in Report Parameter chapter.
- **Data Source** We can fill a Jasper file from a range of datasources like an SQL query, an XML file, a csv file, an HQL (Hibernate Query Language) query, a collection of Java Beans, etc. This will be discussed in detail in Report Data Sources chapter.

The output generated of this process **.jrprint** is a document ready to be viewed, printed or exported to other formats. The facade class *net.sf.jasperreports.engine.JasperFillManager* is usually used for filling a report template with data. This class has various *fillReportXXX()* methods that fill report templates (templates could be located on disk, picked from input streams, or are supplied directly as in-memory).

There are two categories of fillReportXXX() methods in this facade class:

1. The first type, receive a java.sql.Connection object as the third parameter. Most of the times reports are filled with data from a relational database. This is achieved by:
    - Connect to the database through JDBC.
    - Include an SQL query inside the report template.
    - JasperReports engine uses the connection passed in and executes the SQL query.
    - A report data source is thus produced for filling the report.
2. The second type, receive a net.sf.jasperreports.engine.JRDataSource object, when data to be filled is available in other forms.

Filling Report Templates

Let's write a report template. The contents of the JRXML file (C:\tools\jasperreports-5.0.1\test\jasper_report_template.jrxml) are as below:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE jasperReport PUBLIC "//JasperReports//DTD Report Design//EN"
"http://jasperreports.sourceforge.net/dtds/jasperreport.dtd">
<jasperReport xmlns="http://jasperreports.sourceforge.net/jasperreports"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://jasperreports.sourceforge.net/jasperreports
http://jasperreports.sourceforge.net/xsd/jasperreport.xsd"
name="jasper_report_template" language="groovy" pageWidth="595"
pageHeight="842" columnWidth="555" leftMargin="20" rightMargin="20"
topMargin="20" bottomMargin="20">

    <queryString>
 <![CDATA[]]>
    </queryString>
    <field name="country" >
        <fieldDescription><![CDATA[country]]></fieldDescription>
    </field>
    <field name="name" >
        <fieldDescription><![CDATA[name]]></fieldDescription>
    </field>
    <columnHeader>
     <band height="23">
 <staticText>
     <reportElement mode="Opaque" x="0" y="3" width="535"
   height="15" backcolor="#70A9A9" />
     <box>
         <bottomPen lineWidth="1.0" lineColor="#CCCCCC" />
     </box>
     <textElement />
     <text><![CDATA[]]> </text>
 </staticText>
 <staticText>
     <reportElement x="414" y="3" width="121" height="15" />
     <textElement textAlignment="Center"
         verticalAlignment="Middle">
  <font isBold="true" />
     </textElement>
     <text><![CDATA[Country]]></text>
 </staticText>
 <staticText>
     <reportElement x="0" y="3" width="136" height="15" />
     <textElement textAlignment="Center"
         verticalAlignment="Middle">
  <font isBold="true" />
     </textElement>
     <text><![CDATA[Name]]></text>
 </staticText>
     </band>
    </columnHeader>
    <detail>
        <band height="16">
  <staticText>
      <reportElement mode="Opaque" x="0" y="0" width="535"
   height="14" backcolor="#E5ECF9" />
      <box>
              <bottomPen lineWidth="0.25" lineColor="#CCCCCC" />
      </box>
      <textElement />
      <text><![CDATA[]]> </text>
  </staticText>
  <textField>
      <reportElement x="414" y="0" width="121" height="15" />
      <textElement textAlignment="Center"
          verticalAlignment="Middle">
   <font size="9" />
      </textElement>
      <textFieldExpression >
          <![CDATA[$F{country}]]>
  </textFieldExpression>
  </textField>
  <textField>
      <reportElement x="0" y="0" width="136" height="15" />
      <textElement textAlignment="Center"
          verticalAlignment="Middle" />
          <textFieldExpression >
   <![CDATA[$F{name}]]>
          </textFieldExpression>
  </textField>
        </band>
    </detail>
</jasperReport>
```

Next, let's pass a collection of Java data objects (Java beans), to the Jasper Report Engine, to fill this compiled report.

Write a POJO DataBean.java which represents the data object (Java bean). This class defines two String objects name and country. Save it to directory **C:\tools\jasperreports-5.0.1\test\src\com\tutorialspoint**.

```
package com.tutorialspoint;
```

```
public class DataBean {
   private String name;
   private String country;

   public String getName() {
      return name;
   }

   public void setName(String name) {
      this.name = name;
   }

   public String getCountry() {
      return country;
   }

   public void setCountry(String country) {
      this.country = country;
   }
}
```

Write a class DataBeanList.java which has business logic to generate a collection of java bean objects. This is further passed to the Jasper report engine, to generate the report. Here we are adding 4 DataBean objects in the List. Save it to directory **C:\tools\jasperreports-5.0.1\test\src\com\tutorialspoint**.

```
package com.tutorialspoint;

import java.util.ArrayList;

public class DataBeanList {
   public ArrayList<DataBean> getDataBeanList() {
      ArrayList<DataBean> dataBeanList = new ArrayList<DataBean>();

      dataBeanList.add(produce("Manisha", "India"));
      dataBeanList.add(produce("Dennis Ritchie", "USA"));
      dataBeanList.add(produce("V.Anand", "India"));
      dataBeanList.add(produce("Shrinath", "California"));

      return dataBeanList;
   }

   /**
    * This method returns a DataBean object,
    * with name and country set in it.
    */
   private DataBean produce(String name, String country) {
      DataBean dataBean = new DataBean();
      dataBean.setName(name);
      dataBean.setCountry(country);
      return dataBean;
   }
}
```

Write a main class file **JasperReportFill.java**, which gets the java bean collection from the class (DataBeanList) and passes it to the Jasper report engine, to fill the report template. Save it to directory **C:\tools\jasperreports-5.0.1\test\src\com\tutorialspoint**.

```
package com.tutorialspoint;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;

import net.sf.jasperreports.engine.JRException;
import net.sf.jasperreports.engine.JasperFillManager;
import net.sf.jasperreports.engine.data.JRBeanCollectionDataSource;
```

```
public class JasperReportFill {
   @SuppressWarnings("unchecked")
   public static void main(String[] args) {
      String sourceFileName =
         "c://tools/jasperreports-5.0.1/test/jasper_report_template.jasper";
      DataBeanList DataBeanList = new DataBeanList();
      ArrayList<DataBean> dataList = DataBeanList.getDataBeanList();

      JRBeanCollectionDataSource beanColDataSource =
      new JRBeanCollectionDataSource(dataList);

      Map parameters = new HashMap();
      try {
         JasperFillManager.fillReportToFile(
         sourceFileName,
         parameters,
         beanColDataSource);
      } catch (JRException e) {
         e.printStackTrace();
      }
   }
}
```

Generating Reports

We will now compile and execute these files using our regular ANT build process. The build.xml file is as below:

> *The import file - baseBuild.xml is picked from chapter Environment Setup and should be placed in the same directory as the build.xml.*

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="JasperReportTest" default="executereport" basedir=".">
   <import file="baseBuild.xml"/>

   <target name="executereport" depends="compile,compilereportdesing,run">
        <echo message="Im here"/>
   </target>
    <target name="compilereportdesing"
         description="Compiles the JXML file and
         produces the .jasper file.">
         <taskdef name="jrc"
         classname="net.sf.jasperreports.ant.JRAntCompileTask">
            <classpath ref />
         </taskdef>
         <jrc destdir=".">
            <src>
            <fileset dir=".">
               <include name="*.jrxml" />
            </fileset>
            </src>
            <classpath ref />
         </jrc>
   </target>
</project>
```

Next, let's open command line window and go to the directory where build.xml is placed. Finally execute the command **ant -Dmain-class=com.tutorialspoint.JasperReportFill** (**executereport** is the default target) as follows:

```
C:\tools\jasperreports-5.0.1\test>ant -Dmain-class=com.tutorialspoint.JasperReportFill
Buildfile: C:\tools\jasperreports-5.0.1\test\build.xml

compile:
    [javac] C:\tools\jasperreports-5.0.1\test\baseBuild.xml:27:
    warning: 'includeantruntime' was not set, defaulting to
    build.sysclasspath=last; set to false for repeatable builds
    [javac] Compiling 1 source file to
    C:\tools\jasperreports-5.0.1\test\classes

run:
     [echo] Runnin class : com.tutorialspoint.JasperReportFill
     [java] log4j:WARN No appenders could be found for logger
     (net.sf.jasperreports.extensions.ExtensionsEnvironment).
     [java] log4j:WARN Please initialize the log4j system properly.

BUILD SUCCESSFUL
Total time: 8 seconds
```

As a result of above execution a file *jasper_report_template.jrprint* is generated in the same directory as the *.jasper* file (In this case it is generated at C:\tools\jasperreports-5.0.1\test).